
R.C. Chakinala, A. Kumarasubramanian, Kofi A. Laing
R. Manokaran, C. Pandu Rangan, R. Rajaraman
Motivating Application:
Road Traffic Sensor Network Monitoring

- **Periodic Updates**: Road Sensors periodically measure traffic
- **Snapshot Requirement**: Traffic Maps
- **Different Source/Server Periods**: Better Sensors for more important roads
- **Different Sink/Client Periods**: Different Frequency Users (eg. Yahoo Directions vs. Cabbie vs Commuter)
Motivating Application: Road Traffic Sensor Network Monitoring

- **Periodic Updates**: Road Sensors periodically measure traffic
- **Snapshot Requirement**: Traffic Maps
- **Different Source/Server Periods**: Better Sensors for more important roads
- **Different Sink/Client Periods**: Different Frequency Users (eg. Yahoo Directions vs. Cabbie vs Commuter)

Question: Best way to get information updates from sources to sinks?
Push and Pull

Source

Sink
Push and Pull

Source

Push

Sink
Push and Pull

Push

Source

Sink

push
Push and Pull

Push

Sink

Pull

Source
Push and Pull

Push

Source

Sink

color: #666666

Push

Pull

Source

Sink

color: #666666

color: #000000

query
Push and Pull

Push

Source ➔ Sink

Push

Pull

Source ← Sink

Pull
Push and Pull

Push

Source → Sink
push

Pull

Source ← Sink
query
response

Mixed

Source → Store

Sink

Store
Push and Pull

**Push**
- Source
- Sink
- Push

**Pull**
- Source
- Sink
- Pull
- Query
- Response

**Mixed**
- Source
- Sink
- Mixed
- Store
Push and Pull

**Push**

Source → Sink

**Pull**

Source ← Sink

**Mixed**

Source → Store → Sink

- Push
  - push
- Pull
  - query
  - response
- Mixed
  - Store
A Simple Example

Using average source and sink frequencies.

Sources

\[ x \]

Sink

\[ y \]

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

**Push**

<table>
<thead>
<tr>
<th>Sources</th>
<th>Sink</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$y$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Push</th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>$4x$</td>
<td>$4x$</td>
</tr>
<tr>
<td>Pull</td>
<td>$6y$</td>
<td>$7y$</td>
</tr>
<tr>
<td>Mixed</td>
<td>$2x + 2y$</td>
<td>$2x + 3y$</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>$2y &lt; 2x &lt; 4y$</td>
<td>$3y &lt; 2x &lt; 4y$</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

**Push**

<table>
<thead>
<tr>
<th>Sources</th>
<th>Sink</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$y$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>$4x$</td>
<td>$4x$</td>
</tr>
<tr>
<td>Pull</td>
<td>$6y$</td>
<td>$7y$</td>
</tr>
<tr>
<td>Mixed</td>
<td>$2x + 2y$</td>
<td>$2x + 3y$</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>$2y &lt; 2x &lt; 4y$</td>
<td>$3y &lt; 2x &lt; 4y$</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

**Push**

<table>
<thead>
<tr>
<th>Sources</th>
<th>Sink</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>x</code></td>
<td><code>y</code></td>
</tr>
</tbody>
</table>

**Aggregation**

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

Pull

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>$4x$</td>
<td>$4x$</td>
</tr>
<tr>
<td>Pull</td>
<td>$6y$</td>
<td>$7y$</td>
</tr>
<tr>
<td>Mixed</td>
<td>$2x + 2y$</td>
<td>$2x + 3y$</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>$2y &lt; 2x &lt; 4y$</td>
<td>$3y &lt; 2x &lt; 4y$</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

**Pull**

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

**Pull**

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

Pull

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

**Pull**

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

**Pull**

<table>
<thead>
<tr>
<th>Sources</th>
<th>Sink</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$y$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>$4x$</td>
<td>$4x$</td>
</tr>
<tr>
<td>Pull</td>
<td>$6y$</td>
<td>$7y$</td>
</tr>
<tr>
<td>Mixed</td>
<td>$2x + 2y$</td>
<td>$2x + 3y$</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>$2y &lt; 2x &lt; 4y$</td>
<td>$3y &lt; 2x &lt; 4y$</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

**Mixed**

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

Mixed

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

Mixed

<table>
<thead>
<tr>
<th>Sources</th>
<th>Sink</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$y$</td>
</tr>
</tbody>
</table>

| Store   | $y$  |

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>$4x$</td>
<td>$4x$</td>
</tr>
<tr>
<td>Pull</td>
<td>$6y$</td>
<td>$7y$</td>
</tr>
<tr>
<td>Mixed</td>
<td>$2x + 2y$</td>
<td>$2x + 3y$</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>$2y &lt; 2x &lt; 4y$</td>
<td>$3y &lt; 2x &lt; 4y$</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

### Mixed

<table>
<thead>
<tr>
<th></th>
<th>Aggregation</th>
<th>No Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>4x</td>
<td>4x</td>
</tr>
<tr>
<td>Pull</td>
<td>6y</td>
<td>7y</td>
</tr>
<tr>
<td>Mixed</td>
<td>2x + 2y</td>
<td>2x + 3y</td>
</tr>
<tr>
<td>Mixed is Best</td>
<td>2y &lt; 2x &lt; 4y</td>
<td>3y &lt; 2x &lt; 4y</td>
</tr>
</tbody>
</table>
A Simple Example

Using average source and sink frequencies.

Mixed

<table>
<thead>
<tr>
<th>Sources</th>
<th>Sink</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$x$</td>
</tr>
<tr>
<td>$x$</td>
<td>$x$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixed</th>
<th>$2x + 2y$</th>
<th>$2x + 3y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed is Best</td>
<td>$2y &lt; 2x &lt; 4y$</td>
<td>$3y &lt; 2x &lt; 4y$</td>
</tr>
</tbody>
</table>
General Problem — High Level

- **INPUTS:** Graph $G = (V, E)$ with:
  - cost of updating set of stores: $SetC : V \times \text{Powerset}(V) \rightarrow \mathbb{R}^+$
  - Source Set $\mathcal{P} \subseteq V$, Sink Set $\mathcal{Q} \subseteq V$
  - For every source $i \in \mathcal{P}$, a source frequency $p_i$
  - For every sink $j \in \mathcal{Q}$, a sink frequency $q_j$
  - For every sink $j \in \mathcal{Q}$, an interest set $I_j$
General Problem — High Level

• **INPUTS:** Graph $G = (V, E)$ with:
  * cost of updating set of stores: $\text{SetC} : V \times \text{Powerset}(V) \rightarrow \mathbb{R}^+$
  * **Source Set** $\mathcal{P} \subseteq V$, **Sink Set** $\mathcal{Q} \subseteq V$
  * For every source $i \in \mathcal{P}$, a **source frequency** $p_i$
  * For every sink $j \in \mathcal{Q}$, a **sink frequency** $q_j$
  * For every sink $j \in \mathcal{Q}$, an **interest set** $I_j$

• **OUTPUTS:**
  * For every source $i \in \mathcal{P}$, a **Push set** $P_i$
  * For every sink $j \in \mathcal{Q}$, a **Pull Set** $Q_j$
  * Intersection requirement: $i \in I_j \Rightarrow P_i \cap Q_j \neq \emptyset$.
  * **MINIMIZE:** total cost of push-updates, queries and responses:

$$
\sum_{i \in \mathcal{P}} p_i \cdot \text{SetC}(i, P_i) + \sum_{j \in \mathcal{Q}} q_j \cdot \text{SetC}(j, Q_j) + \sum_{j \in \mathcal{Q}} q_j \cdot \text{RespC}(j)
$$
Routing Cost Models

**Multicast**

<table>
<thead>
<tr>
<th>Cost</th>
<th>Example</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multicast</td>
<td>3</td>
<td>Steiner tree cost</td>
</tr>
<tr>
<td>Unicast</td>
<td>4</td>
<td>Sum of path costs</td>
</tr>
<tr>
<td>(non-)metric</td>
<td></td>
<td>Distance function</td>
</tr>
<tr>
<td>Broadcast Model</td>
<td>5</td>
<td>Breadth first tree cost to depth $r$</td>
</tr>
</tbody>
</table>
## Routing Cost Models

### Unicast

<table>
<thead>
<tr>
<th>Cost</th>
<th>Example</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multicast</td>
<td>3</td>
<td>Steiner tree cost</td>
</tr>
<tr>
<td>Unicast</td>
<td>4</td>
<td>Sum of path costs (non-)metric Distance function</td>
</tr>
<tr>
<td>Broadcast Model</td>
<td>5</td>
<td>Breadth first tree cost to depth ( r )</td>
</tr>
</tbody>
</table>
Routing Cost Models

Controlled Broadcast

<table>
<thead>
<tr>
<th>Cost</th>
<th>Example</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multicast</td>
<td>3</td>
<td>Steiner tree cost</td>
</tr>
<tr>
<td>Unicast</td>
<td>4</td>
<td>Sum of path costs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(non-)metric Distance function</td>
</tr>
<tr>
<td>Broadcast Model</td>
<td>5</td>
<td>Breadth first tree cost to depth $r$</td>
</tr>
</tbody>
</table>
Related Work

- FeedTree: RSS via P2P Multicast, [Sandler et al., IPTPS’05]
- Web Caching applications
- Combs, Needles and Haystacks Paper, [Liu et al. SENSYS’04]
- Data Gerrymandering, [Bagchi et al. T.A. TKDE]
- Minimum Cost 2-spanners: [Dodis & Khanna STOC’99] and [Kortsarz & Peleg SICOMP’98]
- Multicommodity facility location, [Ravi & Sinha SODA’04]
- Classical Theory Problems
  - Facility Location
  - Steiner Tree (including Group Steiner Tree)
Our Results

• Multicast Model
  * Exact Tree Algorithm (Distributed)
  * General Graphs
    † $O(\log n)$-Approximation
    † NP-Completeness

• Unicast Model
  * Nonmetric Case — $O(\log n)$-Approximation
  * Identical Interest Sets / Metric Case — $O(1)$-Approximation
  * NP-Completeness

• Controlled Broadcast Model
  * A Polynomial LP solution
  * A Combinatorial solution
The Multicast Model – With Aggregation

- want the following
  - A push subtree $T_i$ for each source $i$
  - A pull subtree $T'_j$ for each sink $j$
  - Whenever $j$ is interested in $i$ ($i \in I_j$), $T_i \cap T'_j \neq \emptyset$.
  - Total cost of all trees (summing edge weights in each tree) is minimized.

- For Trees:
  - Basic idea: for each edge, compute minimum possible cost for connectedness of trees.
  - Claim: Global optimum consists of this solution at every edge.
Interest sets: \{x, z\} want \{a, b, c\}; y only wants a.
Interest sets: \( \{x, z\} \) want \( \{a, b, c\} \); \( y \) only wants \( a \).
• Interest sets: \( \{x, z\} \) want \( \{a, b, c\} \); \( y \) only wants \( a \).

• Question: What is the \textbf{minimum} we can pay on edge \( vw \)?
• Interest sets: \(\{x, z\}\) want \(\{a, b, c\}\); \(y\) only wants \(a\).

• Question: What is the minimum we can pay on edge \(vw\)?
Bipartite Minimum Weight Vertex Cover
Bipartite Minimum Weight Vertex Cover

- **Well Known:** For bipartite $G_{vw} = (A \cup B, E)$, MWVC $\in P$ (Max flow). Find min cut $R$, to get MWVC $C_{vw} = (A \setminus R) \cup (B \cap R)$
• **Well Known:** For bipartite $G_{vw} = (A \cup B, E)$, MWVC $\in P$ (Max flow). Find min cut $R$, to get MWVC $C_{vw} = (A \setminus R) \cup (B \cap R)$
• **Well Known:** For bipartite $G_{vw} = (A \cup B, E)$, MWVC $\in P$ (Max flow). Find min cut $R$, to get MWVC $C_{vw} = (A \setminus R) \cup (B \cap R)$
Bipartite Minimum Weight Vertex Cover

Well Known: For bipartite $G_{vw} = (A \cup B, E)$, MWVC $\in P$ (Max flow). Find min cut $R$, to get MWVC $C_{vw} = (A \setminus R) \cup (B \cap R)$
• **Well Known:** For bipartite $G_{vw} = (A \cup B, E)$, MWVC $\in P$ (Max flow). Find min cut $R$, to get MWVC $C_{vw} = (A \setminus R) \cup (B \cap R)$

• **Application:** Set $A = P_{vw}$ and $B = Q_{vw}$. 
Bipartite Minimum Weight Vertex Cover

- **Well Known:** For bipartite $G_{vw} = (A \cup B, E)$, MWVC $\in P$ (Max flow). Find min cut $R$, to get MWVC $C_{vw} = (A \setminus R) \cup (B \cap R)$

- **Application:** Set $A = P_{vw}$ and $B = Q_{vw}$.

**Note:** For Nonaggregation: set $A = P_{vw}$ and $B = Q_{vw} \cup \{x_{ij} \mid (i, j) \in X_{vw}\}$ for response costs.
**Bipartite Minimum Weight Vertex Cover**

- **Well Known:** For bipartite $G_{vw} = (A \cup B, E)$, MWVC $\in P$ (Max flow). Find min cut $R$, to get MWVC $C_{vw} = (A \setminus R) \cup (B \cap R)$

- **Application:** Set $A = P_{vw}$ and $B = Q_{vw}$.

  **Note:** For Nonaggregation, set $A = P_{vw}$ and $B = Q_{vw} \cup \{x_{ij} \mid (i, j) \in X_{vw}\}$ for response costs.

**Lemma 1.** For each arc $e = vw$, the MWVC weight of $G_{vw}$ is the minimum value paid for $vw$ in any optimal solution.
Interest sets — recall: $\{x, z\}$ want $\{a, b, c\}$; $y$ only wants $a$. 
• Interest sets — recall: \( \{x, z\} \) want \( \{a, b, c\} \); \( y \) only wants \( a \).

• What about \( G_{uv} \)? Clearly different.
• Interest sets — recall: \( \{x, z\} \) want \( \{a, b, c\} \); \( y \) only wants \( a \).

• What about \( G_{uv} \)? Clearly different.
• Interest sets — recall: \( \{x, z\} \) want \( \{a, b, c\} \); \( y \) only wants \( a \).

• What about \( G_{uv} \)? Clearly different.
• Interest sets — recall: \( \{x, z\} \) want \( \{a, b, c\} \); \( y \) only wants \( a \).

• What about \( G_{uv} \)? Clearly different.

• Are push trees, pull trees and response paths connected?
Interest sets — recall: \( \{x, z\} \) want \( \{a, b, c\} \); \( y \) only wants \( a \).

What about \( G_{uv} \)? Clearly different.

Are push trees, pull trees and response paths connected?

**Defn:** In bipartite \( G = (A \cup B, E) \), an MWVC is **\( A \)-maximum** if it has maximum weight in \( A \).
Structural Continuity Solution
Structural Continuity Solution
Structural Continuity Solution
Structural Continuity Solution
Lemma 2. Let $uvw$ be two consecutive edges, let $A$ be the set of push nodes in $G_{uv}$, and let $B$ be the set of (non-push) nodes in $G_{vw}$. 
Lemma 2. Let $uvw$ be two consecutive edges, let $A$ be the set of push nodes in $G_{uv}$, and let $B$ be the set of (non-push) nodes in $G_{vw}$. Let

- $A_1, B_1$ be parts of push-maximum MWVC of $G_{vw}$ in $A, B$ resp., and
Lemma 2. Let $uvw$ be two consecutive edges, let $A$ be the set of push nodes in $G_{uv}$, and let $B$ be the set of (non-push) nodes in $G_{vw}$. Let

- $A_1, B_1$ be parts of push-maximum MWVC of $G_{vw}$ in $A, B$ resp., and
- $A_2, B_2$ be parts of push-maximum MWVC of $G_{uv}$ in $A, B$ resp.
Lemma 2. Let $uvw$ be two consecutive edges, let $A$ be the set of push nodes in $G_{uv}$, and let $B$ be the set of (non-push) nodes in $G_{vw}$. Let
- $A_1, B_1$ be parts of push-maximum MWVC of $G_{vw}$ in $A, B$ resp., and
- $A_2, B_2$ be parts of push-maximum MWVC of $G_{uv}$ in $A, B$ resp.
- then $A_1 \subseteq A_2$ and $B_1 \supseteq B_2$.

Push/Pull subtrees, Response paths are connected!
Tree Algorithm

for each directed edge $uv$
    construct the graph $G_{uv}$
    find its canonical minimum cut $C_{uv}$
    for all $i \in P_{uv}$
        if $i \in C_{uv}$ then include $uv$ in $T_i$
    for all $j \in Q_{vu}$
        if $j \in C_{uv}$ then include $uv$ in $T'_j$
    for all $(i, j) \in X_{uv}$
        if $x_{ij} \in C_{uv}$ then include $uv$ in $P(T_i, j)$
Distributed Implementation

- Global **All-to-all** exchange of
  - sets of push nodes’ frequencies,
  - pull nodes’ frequencies and interest sets.
- Locally, each edge solves both its directions **independently**.
- Use the solution to push and pull information

**Notes:**

- Cost of first phase small compared to third.
- For small sets of distinct values, communication improved.
Unicast Model with Aggregation
An Integer Program

- Replace response cost by doubling sink frequencies
- \( x_{ik} = 1 \) means \( i \) pushes to \( k \)
- \( y_{kj} = 1 \) means \( j \) pulls from \( k \)
- \( r_{ijk} = 1 \) means \( i \) talks to \( j \) through \( k \).
Unicast Model with Aggregation
An Integer Program

- Replace response cost by doubling sink frequencies
- $x_{ik} = 1$ means $i$ pushes to $k$
- $y_{kj} = 1$ means $j$ pulls from $k$
- $r_{ijk} = 1$ means $i$ talks to $j$ through $k$.

Minimize: \[ \sum_{i \in P} p_i \sum_{k \in V} d_{ik} x_{ik} + \sum_{j \in Q} q_j \sum_{k \in V} d_{kj} y_{kj} \]

subject to \[ \begin{cases} r_{ijk} \leq x_{ik} \\ r_{ijk} \leq y_{kj} \\ \sum_k r_{ijk} \geq 1 \end{cases}, \text{ where } x_{ik}, y_{kj}, r_{ijk} \in \{0, 1\} \]
Unicast Model with Aggregation
Uniform Interests, Metric Case — $O(1)$-Approximation

- **Overview**
  - Applies for Identical/Disjoint Interest Sets
  - Uses same Integer Program.
  - Deterministic Rounding with Filtering Technique Lin & Vitter IPL’92, Shmoys et al STOC’97, Ravi & Sinha SODA’04
Basic definitions

- Optimal solution to the LP is \((x^*, y^*, r^*)\).
- LP gives cost lower bounds \(C_i = \sum_k d_{ik}x_{ik}^*\) and \(C'_j = \sum_k d_{kj}y_{kj}^*\)
Basic definitions

* Optimal solution to the LP is \((x^*, y^*, r^*)\).
* LP gives cost lower bounds \(C_i = \sum_k d_{ik} x^*_{ik}\) and \(C'_j = \sum_k d_{kj} y^*_{kj}\).
* For node \(u, r > 0\), define \(B_u(r) = \{v : d_{uv} \leq r\}\).
* Let \(1 < \alpha < \beta\). Clearly \(B_j(C'_j) \subseteq B_j(\alpha C'_j) \subseteq B_j(\beta C'_j)\).
Unicast Model with Aggregation
Uniform Interest Set / Metric — Algorithm

- Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Unicast Model with Aggregation
Uniform Interest Set / Metric — Algorithm

• Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Unicast Model with Aggregation
Uniform Interest Set / Metric — Algorithm

- Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Unicast Model with Aggregation
Uniform Interest Set / Metric — Algorithm

- Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
Unicast Model with Aggregation
Uniform Interest Set / Metric — Algorithm

• Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.

• Define push sets: $P_i = \{i\} \cup \{\ell_i\} \cup \{j : j \in S' \text{ and } C'_j \leq C_i\}$ and pull sets: $Q_j = \{j\} \cup \{\ell'_j\} \cup \{i : i \in S \text{ and } C_i < C'_j\}$. 
Unicast Model with Aggregation

Uniform Interest Set / Metric — Algorithm

- Choose leaders: nodes with disjoint $\beta$-balls, by nondecreasing cost.
- Define push sets: $P_i = \{i\} \cup \{\ell_i\} \cup \{j : j \in S' \text{ and } C_j' \leq C_i\}$
  and pull sets: $Q_j = \{j\} \cup \{\ell_j'\} \cup \{i : i \in S \text{ and } C_i < C_j'\}$.
- Intersection guarantee: For each $i \in P$ and $j \in Q$, $P_i \cap Q_j \neq \emptyset$. 

Diagram:

- Nodes are connected with arrows indicating the direction of data flow.
- Leaders are marked with red circles.
- Nodes $m$, $i$, and $j$ are highlighted for emphasis.
Unicast Model with Aggregation
Uniform Interest Set / Metric — Algorithm Proof

- Relative distance limits total push extent:
  For $i \in \mathcal{P}$, $\alpha > 1$, \[ \sum_{k \notin B_i(\alpha C_i)} x_{ik}^* \leq 1/\alpha \]
Unicast Model with Aggregation
Uniform Interest Set / Metric — Algorithm Proof

- Relative distance limits total push extent:
  For \( i \in \mathcal{P}, \alpha > 1, \sum_{k \notin B_i(\alpha C_i)} x^*_i \leq 1/\alpha \)

- Derive Approximation Ratio.
  - Recall: \( P_i = \{i\} \cup \{\ell_i\} \cup \{j : j \in S' \text{ and } C'_j \leq C_i\} \)
  - Cost to \( i \)'s leader \( \ell_i \): \( 2\beta C_i \)
Unicast Model with Aggregation
Uniform Interest Set / Metric — Algorithm Proof

• Relative distance limits total push extent:
For $i \in \mathcal{P}$, $\alpha > 1$, $\sum_{k \notin B_i(\alpha C_i)} x_{ik}^* \leq 1/\alpha$

• Derive Approximation Ratio.
  * Recall: $P_i = \{i\} \cup \{\ell_i\} \cup \{j : j \in S' \text{ and } C'_j \leq C_i\}$
  * Cost to $i$’s leader $\ell_i$: $2\beta C_i$
  * Cost to (other) leaders $S_i$:
    $$C_i \geq \sum_{j \in S_i} (d_{ij} - \alpha C'_j) \sum_{k \in B_j(\alpha C'_j)} r_{ijk}^*$$
Unicast Model with Aggregation

Uniform Interest Set / Metric — Algorithm Proof

• Relative distance limits total push extent:
  For \( i \in \mathcal{P}, \alpha > 1, \sum_{k \notin B_i(\alpha C_i)} x_{ik}^* \leq \frac{1}{\alpha} \)

• Derive Approximation Ratio.
  \* Recall: \( P_i = \{i\} \cup \{\ell_i\} \cup \{j : j \in S' \text{ and } C_j' \leq C_i\} \)
  \* Cost to \( i \)'s leader \( \ell_i \): \( 2\beta C_i \)
  \* Cost to (other) leaders \( S_i \):

\[
C_i \geq \sum_{j \in S_i} (d_{ij} - \alpha C_j') \sum_{k \in B_j(\alpha C_j')} r_{ijk}^*
\geq \sum_{j \in S_i} d_{ij} \left[ 1 - \frac{\alpha}{\beta} \right] \left[ 1 - \frac{1}{\alpha} \right]
= \frac{(\beta - \alpha)(\alpha - 1)}{\alpha \beta} \sum_{j \in S_i} d_{ij}.
\]

\* \( \alpha = 1.69 \) and \( \beta = 2.86 \) obtains 14.57-approximation.
Conclusions and Open Problems

- Nonuniform Packet Lengths

- Multicast:
  - General Graphs; Can $O(\log n)$ UB be improved to $O(1)$?

- Nonmetric Unicast:
  - Derandomizing $O(\log n)$ algorithm.
  - Close gap $O(1)$ LB vs $O(\log n)$ UB gap

- Metric Unicast Case
  - Improving the 14.57 bound for Uniform Interest sets.
  - Non-uniform interest sets (UB and/or Hardness)

- Dynamic Graphs — Frequency, Position and Topology changes
Thank You!

- **Chakinala, Kumarasubramanian, and Manokaran:** partial support — generous gift from Northeastern University alumnus Madhav Anand.

- **Rajaraman:** partial support — NSF grant IIS-0330201.

- **Laing:** partial support — the Mellon Foundation and the Faculty Research Awards Committee of Tufts University, while visiting Northeastern University.